# Requirements Gathering

## Functional Requirements

* User Authentication & Account Management
* Username – Username field must not be blank.
* Password – Password field must not be blank.
* Credentials – Invalid username/password should produce an appropriate error message.
* Login Process – A valid username and password combination must successfully log in and redirect the user to the secure area.
* Forgot password :  
   Email must be filled

New Password must not be blank

Enter at-least one numeric value

Enter at-least one special character

Choose a difficult Password

Confirm Password must not be blank

Passwords do not Match

### Add/Remove elements

* User is able to add and remove items.

### Checkbox

* Checkbox Presence – The page must display all expected checkboxes (e.g., “Checkbox 1” and “Checkbox 2”).
* Toggle Functionality – Clicking on a checkbox must toggle its state (checked/unchecked) consistently.
* Default State – Predefined default states (e.g., first checkbox unchecked and second checkbox checked) must be working as specified

### Context menu

* Context menu items are custom additions that appear in the right-click menu.

### Disappearing elements

This example demonstrates when elements on a page change by disappearing/reappearing on each page load.

* Home : Move user to the home page
* About : Move user to the about page
* Contact us: Shows the contact menu (emails , phone number , ….etc)
* Portfolio : Moves to the portfolio page

### Drag and drop

* User can drag and drop items in purpose to arrange it to the desired position.

### Dropdown list

* Dropdown Visibility – The Dropdown element should be visible and clickable.
* Option List – All expected options must be present in the dropdown menu.
* Selection – Selecting an option should update the dropdown value accordingly, with no invalid entries accepted.

### Dynamic content

* demonstrates the ever-evolving nature of content by loading new text and images on each page refresh.

### Dynamic controls

demonstrates when elements (e.g., checkbox, input field, etc.) are changed asynchronously.

* If a button is clicked a loading bar will appear to show thar order is in progress and it will disappear when its done

### Dynamic loaded page element

action get triggered that returns a result dynamically. It does not rely on the page to reload or finish loading. The page automatically gets updated through the use of JavaScript.

* element already exists on the page but it is not displayed. And anonther where the element is not on the page and gets added in.

### Entry add and exit intent

* Add appears when you enter the website
* Exits the website when you click on exit intent

### File download and upload

* Upload Button – The “Choose File” button must be visible and functional.
* File Selection – Only files of allowed types (e.g., .txt or .jpg based on scenario) should be selectable.
* Upload Response – The system must display clear confirmation (or error) upon file upload.
* Download Button – The “Choose File” button must be visible and functional.
* Download Response – The system must display clear confirmation (or error) upon file Download.

### Floating menu

* Floating menu appears even if you scrolled down
* Floating menu must be clickable
* floating menu must do an action if clicked

### JavaScript alerts

(This feature tests the appearance and behavior of JavaScript-triggered alerts.)

* Alert Trigger – Clicking the “JS Alert” button must immediately display a JavaScript alert.
* Alert Message – The text on the alert should match the expected message.
* Alert Dismissal – Dismissing the alert (accepting or canceling) should close it without errors

### Infinite scroll

Content Load Trigger:

• When the user scrolls near the bottom of the currently loaded content (e.g., within 10% of the page’s end), the system shall automatically trigger a request to load additional content.

Loading Indicator:

• The system shall display a visible loading animation or message (e.g., a spinner or “Loading…” indicator) during the fetch operation for new content.

Content Appending:

• Upon successful retrieval, the system shall append the newly fetched content to the current content list seamlessly without requiring a page refresh or navigation to another URL.

End-of-Content Notification:

• If no additional content is available beyond the current set, the system shall display a clear message (e.g., “No more items to load”) to inform the user.

Request Management:

• The system shall prevent overlapping or multiple simultaneous fetch requests, ensuring that a new content load request is not initiated until the current request is fully processed.

Browser and Device Compatibility:

• Infinite scroll functionality shall work consistently across supported browsers and devices, ensuring smooth scrolling and content appending regardless of screen size or network speed.

### Key presses

* System shows the last entered element from keyboard

## Non-functional Requirements

* Performance
* All dynamic elements (validation messages, alerts, page responses) should render within 2–3 seconds.
* The site should handle multiple simultaneous test actions without significant slowdown.
* Security
* Input fields must be validated to prevent injections (e.g., in the login forms).
* File uploads must handle validations to avoid uploading malicious files.
* Usability
* Error messages and alerts must be clear, concise, and guide the user toward corrective action.
* The UI should be intuitive, with elements (buttons, fields, menus) accessible on various devices and screen sizes.
* Compatibility
* All features should function consistently across major browsers (Chrome, Firefox, Edge, Safari).
* Responsive design should be verified on mobile and desktop platforms.

## User Stories & Use Cases

### User Story 1 – Form Authentication

• Card (Story Statement):

As a registered user, I want to log in securely so that I can access my personal dashboard and use secure features.

#### • Conversation:

The user will navigate to the login page, enter their username and password, and then click the “Login” button. The system should validate the input; if a field is blank or the credentials are incorrect, an appropriate error message displays. For valid credentials, the user is redirected to the secure area.

#### • Confirmation (Acceptance Criteria):

1. The login page contains input fields for username and password and a "Login" button.

2. If the username or password field is blank, an error is displayed (e.g., “Username is required” or “Password is required”).

3. If an invalid combination is entered, the user sees a message stating “Invalid username or password.”

4. On successful login, the user is directed to their secure dashboard.

### User Story 2 – Using Checkboxes

#### • Card (Story Statement):

As a user, I want to select or unselect checkboxes so that I can customize my choices and filter content as needed.

#### • Conversation:

On the Checkboxes page, users will view multiple checkboxes with default states. Users can toggle checkboxes by clicking on them. The system must reflect the correct state (checked or unchecked) immediately without errors.

#### • Confirmation (Acceptance Criteria):

1. The page displays all expected checkbox elements with defined default states.

2. When a checkbox is clicked, its visual state changes to reflect whether it is checked or unchecked.

3. The action is registered immediately without delay or error.

### User Story 3 – Interacting with the Dropdown

#### • Card (Story Statement):

As a user, I want to select an option from a dropdown menu so that I can filter or modify the view according to my preferences.

#### • Conversation:

The user clicks on the dropdown element to reveal all available options. Upon selecting an option, the dropdown should update to show the chosen value, and relevant content should refresh (if applicable).

#### • Confirmation (Acceptance Criteria):

1. The dropdown is clearly visible and clickable.

2. All expected options are listed when the dropdown is activated.

3. Selecting an option updates the dropdown display accordingly; no invalid selections are allowed.

### User Story 4 – File Upload Process

#### • Card (Story Statement):

As a user, I want to upload a file so that I can share documents or images with the website.

#### • Conversation:

On the File Upload page, the user will see a “Choose File” button. After selecting a file (with allowed file types), the user initiates the upload. The system verifies the file type and size, then provides a clear confirmation or error message upon upload completion.

• Confirmation (Acceptance Criteria):

1. The file upload section contains a clearly labeled “Choose File” button.

2. The system validates the file type before upload (e.g., only .txt and .jpg files allowed).

3. On file selection and upload, the system displays a success message (or an error if the file does not meet criteria).

### User Story 5 – Downloading Files

• Card (Story Statement):  
As a registered user, I want to download files from the website so that I can access necessary documents offline for later review.

• Conversation:  
When the user navigates to a page containing downloadable resources, they will see a clearly labeled “Download” button or link. Clicking this control should initiate a file download process. The system must verify that the requested file exists and is available, then start the download promptly. If any issues occur (e.g., file not found or download error), the system provides clear feedback so that the user understands what went wrong and can take corrective action.

• Confirmation (Acceptance Criteria):  
1. A clear and accessible “Download” button or link is displayed on the relevant page.  
2. Upon user click, the system verifies file availability and initiates the download process immediately.  
3. The downloaded file must be complete, correctly formatted, and match the expected file size and type.  
4. In case of an error during the download (e.g., missing file or network interruption), an appropriate error message is displayed to the user indicating the issue.  
5. The download functionality is tested across supported browsers and devices to ensure consistent performance.

### User Story 6 – Handling JavaScript Alerts

#### • Card (Story Statement):

As a user, I want to receive immediate feedback via JavaScript alerts so that I can be informed of key events or confirmations.

#### • Conversation:

When the user clicks the button that triggers the alert, a JavaScript alert appears with a specific message. The user must acknowledge the alert by clicking “OK” or dismiss it appropriately. The action should be seamless and bring the user back to the previous workflow.

#### • Confirmation (Acceptance Criteria):

1. A click on the “JS Alert” button immediately triggers a JavaScript alert.

2. The alert message matches the expected text defined in the requirements.

3. Dismissing the alert (e.g., clicking “OK”) closes it without error and returns focus to the underlying page.

### User Story 7 – Adding and Removing Elements

#### • Card (Story Statement):

As a user, I want to dynamically add or remove elements from a page so that I can customize the content I see based on my needs.

#### • Conversation:

The user may click an “Add Element” button to create a new element on the page. Similarly, clicking a “Delete” button adjacent to dynamically created elements should remove that specific element instantly without affecting the rest of the page layout.

#### • Confirmation (Acceptance Criteria):

1. Clicking “Add Element” results in the immediate creation of a new element on the page.

2. Each added element has an accessible “Delete” button.

3. Clicking the “Delete” button removes only the corresponding element from the page.

4. The page updates correctly without any errors after adding or removing elements.

### User Story 8 – Dynamic Content Loading

#### • Card (Story Statement):

As a user, I want dynamic content to load seamlessly so that I can instantly view updated information without refreshing the entire page.

#### • Conversation:

When invoking functions that load dynamic content (e.g., click to load more items), a loading indicator appears. Once the content is fetched, the new elements are shown, replacing or supplementing the existing content smoothly.

#### • Confirmation (Acceptance Criteria):

1. A loading indicator is visible when dynamic content is being loaded.

2. New content replaces or is added to the existing content within a reasonable period (e.g., 2–3 seconds).

3. There are no display errors or missing content once the dynamic load is complete.

### User Story 9 – Floating Menu

#### • Card (Story Statement):

As a frequent user, I want the website to display a floating menu so that I can easily navigate to key sections regardless of where I am on the page.

#### • Conversation:

When the user scrolls through the webpage, a floating menu should remain visible on the screen. This menu must not obstruct the main content and should offer direct links to essential sections (such as Home, About, Services, and Contact). The design aims to facilitate quick navigation without requiring the user to scroll back to the top. Developers and testers will review the layout to ensure that the floating menu is both responsive and accessible on various devices and browser sizes.

#### • Confirmation (Acceptance Criteria):

1. The floating menu remains visible while the user scrolls up or down the page.

2. The menu is responsive and maintains its position on different screen sizes and orientations.

3. The menu items link to the correct sections of the website, and those sections are smoothly scrolled into view when selected.

4. The menu does not overlap or obscure important content; its design ensures clear visibility and accessibility of both the menu and the page content.

5. Visual feedback (such as a hover effect) is provided when users interact with the menu items.

6. The menu's performance is verified across major browsers and devices, ensuring consistent behavior.

### User Story 10 – Drag and Drop Functionality

#### • Card (Story Statement):

As a user, I want to drag and drop items on the page so that I can rearrange or organize elements as needed.

#### • Conversation:

On a page that supports drag and drop, the user will click and hold an item (the drag source) and then move it to the desired location (the drop target). The system should visually indicate the drag, highlight the drop target, and reorder or reposition the item upon release. Any rearrangement should be persistent (if applicable) or clearly update the page view instantly.

#### • Confirmation (Acceptance Criteria):

1. The draggable items are clearly marked and can be clicked and held to initiate a drag.

2. When dragging an item, the drop target area is highlighted.

3. Releasing the mouse over a valid drop area moves the item to the new location.

4. The new order of items is accurately updated and persists as expected.

5. If the drop occurs over an invalid area, the item returns to its original position with an appropriate visual cue.

### User Story 11 – Context Menu Interactions

• Card (Story Statement):  
As a user, I want to activate a context menu via right-click so that I can quickly access related actions for a specific element on the page.

• Conversation:  
On relevant elements, the user can right-click to open a context menu containing options like “Edit,” “Delete,” or “Properties.” The user will then select an appropriate action from the menu. The system should display the context menu correctly without interfering with the default page behavior, and selecting an option should trigger the associated functionality.

• Confirmation (Acceptance Criteria):  
1. Right-clicking on target elements triggers the appearance of a context menu.  
2. The context menu displays all intended options clearly and in a user-friendly format.  
3. Selecting an option from the context menu initiates the corresponding action.  
4. The context menu closes automatically upon selection, clicking outside it, or after a set timeout.  
5. No unexpected browser default context menu appears when interacting with the element.

### User Story 12 – Key Presses

#### • Card (Story Statement):

As a user, I want specific key presses to trigger actions on the website so that I can navigate or perform tasks using keyboard shortcuts.

#### • Conversation:

The website will offer keyboard shortcuts for common actions (such as refreshing content, navigating menus, or submitting forms). When the designated key or combination is pressed, the system should perform the respective action without interfering with native browser functionality. Users should have a brief on-screen prompt or visual feedback indicating that the key press was recognized and the action was executed.

#### • Confirmation (Acceptance Criteria):

1. A set of keyboard shortcuts is documented and available to users.

2. Pressing a designated key or key combination triggers the intended action (e.g., “Ctrl + R” refreshes a particular section, “Alt + N” opens a new window) promptly and without error.

3. Visual or auditory feedback confirms the execution of the keyboard command.

4. The assigned key presses do not conflict with browser or system shortcuts.

5. All implemented key press actions are tested across different browsers to ensure consistent behavior.

### User Story 13– Infinite Scroll

#### Card (Story Statement):

As a user, I want the content on the page to load continuously as I scroll down so that I can seamlessly browse more information without manually clicking pagination links.

#### Conversation:

When the user scrolls towards the bottom of a long content list or feed, the system should automatically initiate a request to load additional content. Instead of having to click on a “Next” button or navigating to another page, new content is appended to the existing list. The process should be smooth and should include visual cues (like a loading spinner) indicating that additional content is being fetched. Developers and testers will ensure that the infinite scroll is responsive, loads new content reliably, and does not overload the user’s browser by fetching content too quickly or inefficiently.

#### Confirmation (Acceptance Criteria):

1. When the user nears the bottom of the page, a loading indicator (e.g., a spinner or “Loading…” message) appears.

2. Additional content is fetched and appended to the end of the current list seamlessly and without errors.

3. Scrolling remains smooth without noticeable lag or disruption while new content loads.

4. The process works reliably across different browsers and devices, with no broken layouts or unresponsive behavior.

5. Once new content is loaded, the loading indicator disappears.

6. If no more content is available, the system displays a clear message (e.g., “No more items to load”) so that the user is informed.

### User Story 14 – Overall System Quality (Performance, Security & Usability)

#### • Card (Story Statement):

As a website user, I want the application to be fast, secure, and easy to use so that I can have a seamless and safe experience whenever I visit.

#### • Conversation:

The website must meet non‑functional quality criteria to ensure that user interactions are smooth and secure. This means that pages should load rapidly (even during peak usage), security measures must protect my personal information, and the interface should be intuitive and accessible on various devices. The development and testing teams will work together to define performance thresholds, security parameters (such as encryption and vulnerability protections), and usability guidelines. Regular reviews and metrics will confirm that the system consistently meets these quality measures.

#### • Confirmation (Acceptance Criteria):

1. Performance:

• Each page should load within 3 seconds under standard load conditions.

• The system must handle a specified number of concurrent users without significant degradation in response times.

2. Security:

• The website must use secure protocols (e.g., HTTPS) for data transmission.

• All user inputs will be validated and protected against common vulnerabilities such as SQL injection and XSS attacks.

3. Usability:

• The interface must be clear and consistent, with user testing confirming ease of navigation and accessibility across desktop and mobile devices.

4. Reliability:

• System uptime should be maintained above 99.5% as measured over a given period.

5. Documentation & Monitoring:

• Non‑functional performance, security, and usability metrics must be documented and monitored continuously, with regular reports generated for review.

## expected behavior

1. Form Authentication  
   • When a registered user navigates to the login page, both username and password input fields are visible and clearly labeled.  
   • If either field is left blank, the system displays a prompt (e.g., “Username is required,” “Password is required”).  
   • If invalid credentials are entered, an error message (“Invalid username or password”) is immediately shown.  
   • For valid credentials, the system successfully logs the user in and redirects them to their secure dashboard.
2. Checkbox Interaction  
   • All expected checkboxes are rendered with their default states (checked/unchecked) accurately displayed on page load.  
   • When the user clicks on a checkbox, the system toggles its state instantly and correctly.  
   • The new state of each checkbox is immediately registered and visually updated, ensuring consistent feedback.
3. Dropdown Selection  
   • The dropdown element is visible and clickable, listing all expected options.  
   • When a user clicks the dropdown, all available options appear in a clear and organized list.  
   • Selecting an option updates the dropdown’s displayed value without errors, and any dependent content is refreshed if applicable.
4. File Upload  
   • A clearly labeled “Choose File” button is visible on the upload page.  
   • The system only accepts files of permitted types (as defined by requirements) and verifies file properties such as size.  
   • Once a file is selected, the upload process begins immediately, and a confirmation (or error) message is shown based on the file’s validity and upload success.
5. Downloading Files  
   • Downloadable resources are clearly marked with a “Download” button or link.  
   • Upon clicking the link, the system checks for file availability and immediately begins the download process.  
   • The downloaded file must be complete, correctly formatted, and its size and type must match the defined expectations.  
   • If a download error occurs, a clear error message is provided to help the user understand the problem.
6. JavaScript Alerts  
   • Clicking the designated button for triggering an alert results in the immediate appearance of a JavaScript alert with the expected message.  
   • The alert behaves as intended—after the user acknowledges or dismisses it, the alert closes and focus is returned to the underlying page without residual interference.
7. Adding and Removing Elements  
   • Clicking the “Add Element” button promptly creates a new element on the page, which is visible and immediately available for user interaction.  
   • Each dynamically added element includes a “Delete” button.  
   • Pressing the “Delete” button removes the respective element instantly and without disrupting the layout or other page elements.
8. Dynamic Content Loading  
   • When dynamic content is being requested, a visible loading indicator appears immediately and remains until the new content is rendered.  
   • The new content is displayed within a reasonable time (e.g., 2–3 seconds) without errors, and the transition is smooth.
9. Expected Behavior – Floating Menu  
   • The floating menu is always visible regardless of the page scroll position. As the user scrolls up or down, the menu remains fixed in place.

• The menu is designed responsively; it maintains the correct size, position, and layout on various devices and screen orientations. On smaller screens (e.g., mobile), the menu adapts (possibly collapsing into a hamburger menu) to avoid obscuring content.

• Each menu item is clearly visible, with sufficient spacing and contrast against the background, ensuring that it does not interfere with the readability of the main content.

• When a user clicks on a menu item, the page scrolls smoothly to the corresponding section, with no abrupt jumps or delays.

• Hovering over or focusing on menu items provides immediate visual feedback (such as a change in color, underline, or other styling effects) to indicate interactivity.

• The floating menu does not overlap or hide critical content; it is positioned and styled in a way that both the menu and underlying content are easily accessible and legible.

• Performance is consistent across supported browsers (Chrome, Firefox, Edge, Safari) and devices, with no noticeable lag or misalignment in the menu’s presentation when interacting with the page.

• If the menu contains dynamic elements (e.g., expandable submenus), clicking those elements triggers immediate, smooth animations with appropriate visual cues, without affecting overall page performance.

1. Drag and Drop Functionality  
   • Draggable items are distinctly marked and can be clicked, held, and dragged without delay.  
   • As an item is dragged, the drop target areas become clearly highlighted to guide the user.  
   • Dropping an item in a valid area moves it to the new location with an updated order that is immediately reflected on the page.  
   • If the item is dropped in an invalid area, the system reverts the item to its original position with a visual cue indicating the invalid drop.
2. Context Menu Interactions  
   • Right-clicking on designated elements suppresses the default browser context menu and instead displays a custom context menu with all expected options.  
   • The context menu appears clearly and is easily navigable, showing options like “Edit,” “Delete,” or “Properties.”  
   • Selecting an option from the menu triggers the corresponding action promptly, while the menu then disappears automatically.
3. Key Presses  
   • Specific, pre-defined keyboard shortcuts are available on the site.  
   • When the user presses the designated key or combination (e.g., “Ctrl + R” for refresh, “Alt + N” for new window), the intended action is executed immediately.  
   • Visual or auditory feedback indicates that the key press has been recognized, and actions do not conflict with standard browser shortcuts.  
   • The assigned shortcuts work uniformly across supported browsers and devices.
4. Non-functional Requirements (Overall System Quality)  
   • Overall page responses and dynamic elements are rendered within acceptable performance thresholds, ensuring pages load in under 3 seconds under standard conditions.  
   • All input validations, data transmissions, and file uploads are protected using secure protocols to prevent vulnerabilities (e.g., SQL injection, XSS).  
   • The user interface remains intuitive and accessible across various devices and browsers, with clear error messaging and guidance across every interaction.  
   • The system maintains high reliability and uptime, and regular monitoring confirms that responsiveness and security standards are met consistently.